



Linuxリソース管理機構の AV機器への適用

“Applying the Linux Cgroups Mechanism to AV appliances”

2013年3月8日

三菱電機(株)情報技術総合研究所

遠藤 幸典

1. 背景と目的
2. Linuxリソース管理機構の概要
3. AV機器への適用検討
4. Android環境での適用検証
5. まとめ

1. 背景と目的

- (1) スマホやスマートTV等の登場で、ユーザが自由に多数のアプリをダウンロードして実行できる環境が普及
- (2) 工場出荷後F/W構成や動作組合せは変わらないという従来の組込み機器の前提が変わった！
- (3) 特にAV機器（TV、レコーダ、ナビ、STB等）を考えた時、第三者アプリと共存する環境でも、ユーザが安定して映像を視聴できる仕組みを提供することが重要
- (4) そこで、Linuxリソース管理機構の適用を検討
- (5) AV機能アプリが安定動作可能なソフトウェア基盤を構築すること（方式の検討・検証）を目的とする

2. Linuxリソース管理機構の概要

2.1. Cgroups (Control Groups)とは

- (1)任意のタスクをグループ化して(リソースグループと呼ぶ)制御するためのLinuxカーネルの機能
- (2)Cgroups自体は、グループ化のための機構とI/Fを提供しているだけ
- (3)これをベースにCPU、メモリ、ディスクI/O等に関する具体的なリソース管理機構が実装されており、タスクが所属するリソースグループ毎に監視(Accounting)、優先付け(Prioritization)、制限(Resource Limiting)、及び分離(Isolation)が可能

2. Linuxリソース管理機構の概要

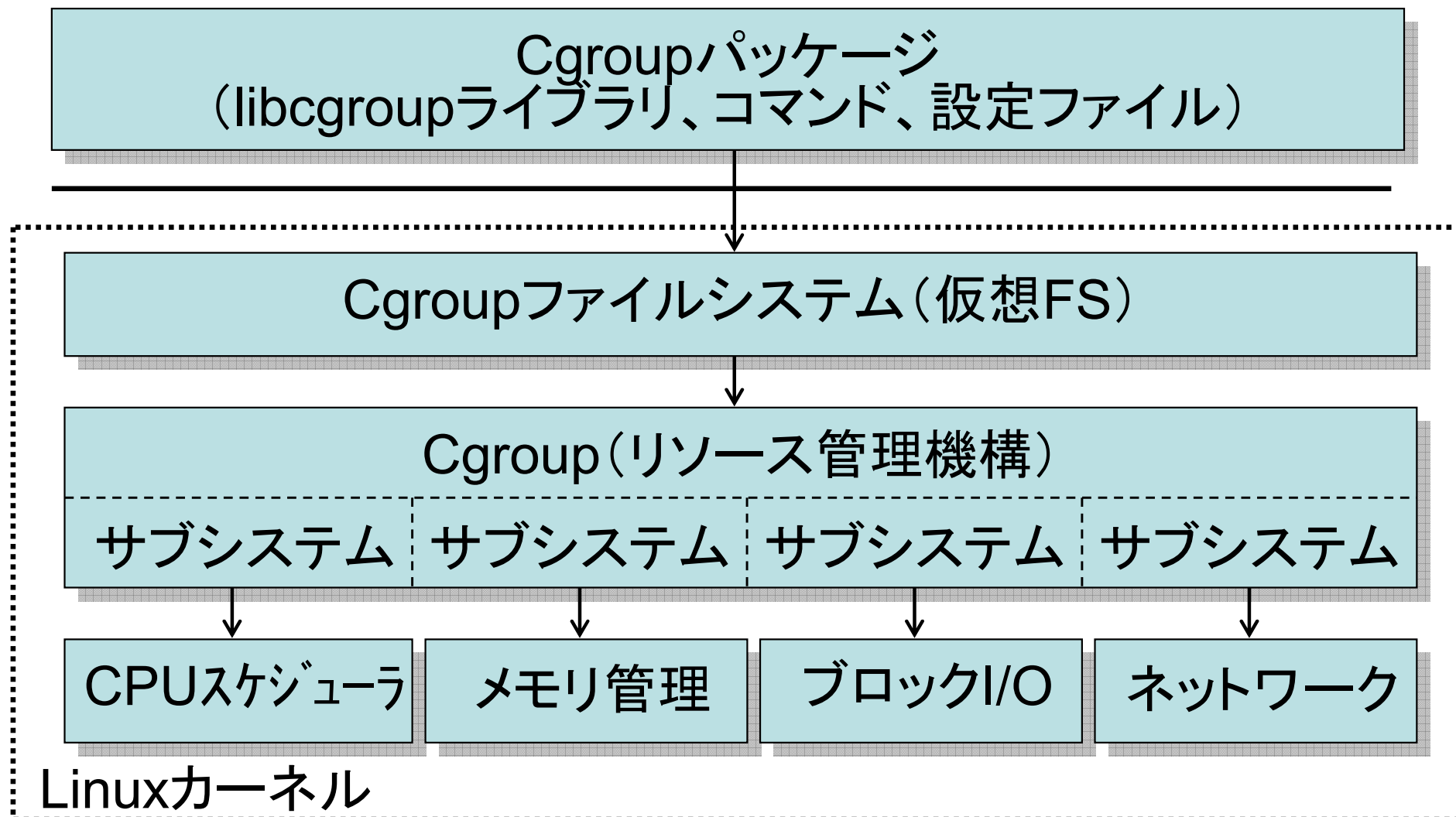
2.2. リソース種別毎のリソース管理機構一覧

※これらを「サブシステム」や「コントローラ」と呼ぶ

| サブシステム名 | リソース管理の内容 |
|---------|------------------------------|
| cpu | CPU使用の優先付け、グループ間の占有率の定義 |
| cpuacct | CPU使用時間の監視、報告 |
| cgroup | マルチCPUの割当て制御 |
| memory | メモリ使用量の制限、報告 |
| blkio | ブロックI/O要求の優先付け、グループ間の占有率の定義 |
| devices | デバイスのアクセス制御 |
| net_cls | ネット通信の優先付け(パケットへのクラスタIDタグ付け) |
| freezer | タスクグループ全体の一時的停止、再開 |

2. Linuxリソース管理機構の概要

2.3. Cgroupsのソフトウェア構成



2.4. Cgroupsの主要履歴

| | |
|----------|---|
| 2006年9月 | Paul Menage氏が”Generic Container system”パッチ投稿 |
| 2008年1月 | Linux 2.6.24でCgroupsマージ(Fair Group Sched導入) |
| 2008年4月 | 同2.6.25でmemoryサブシステムと RT Group Sched導入 |
| 2008年7月 | 同2.6.26でdeviceサブシステム追加 |
| 2008年12月 | 同2.6.28でfreezerサブシステム追加 |
| 2009年3月 | 同2.6.29でmemoryサブシステム改良、pkt_sched追加 |
| 2009年6月 | 同2.6.30でcpuacctサブシステム追加 |
| 2010年2月 | 同2.6.33でblkioサブシステム追加 |
| 2011年1月 | 同2.6.37でblkioサブシステムにI/O throttling(帯域制御)追加 |
| 2011年3月 | 同2.6.38でblkioサブシステム改良 |
| 2012年3月 | 同3.3でnet_prioサブシステム追加 |

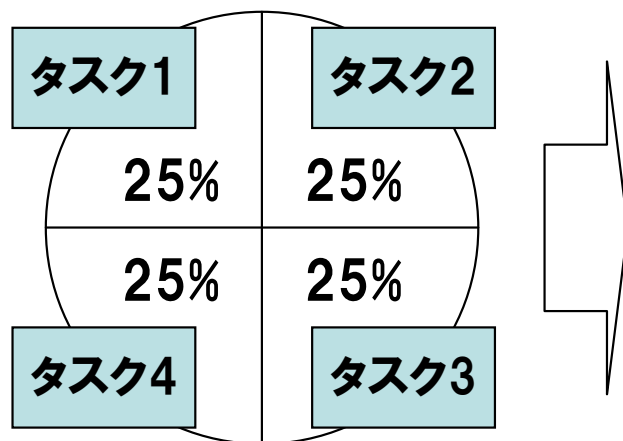
3. AV機器への適用検討

3.1. cpuサブシステム(「Fair Group Scheduling」)

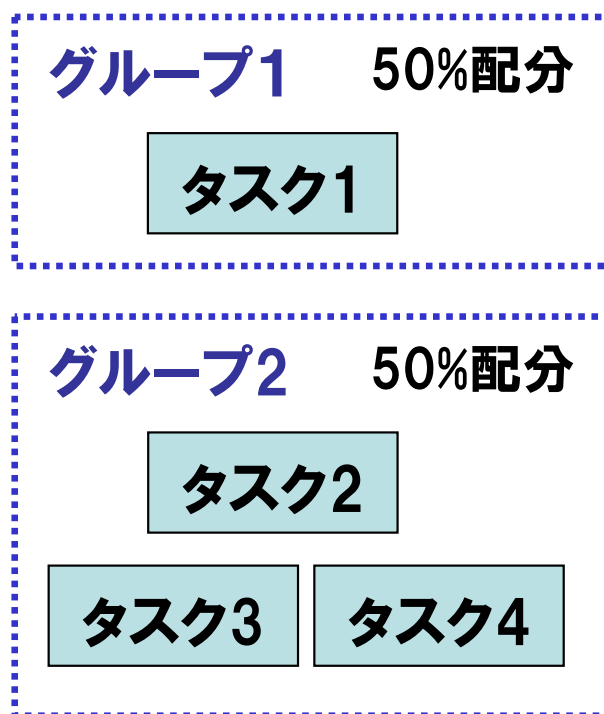
⇒グループ単位のCPU割当て時間の制御を可能にする機構。

⇒優先処理したい通常タスク(Androidアプリ)に、より潤沢なりソース配分をする適用方法が可。

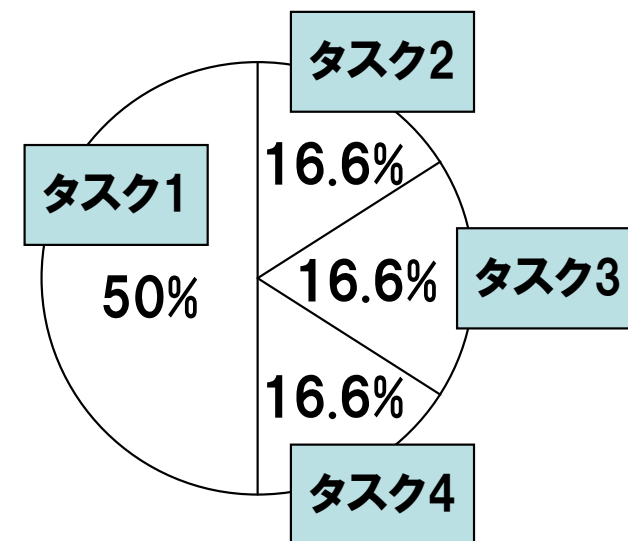
<Before> 適用なし



※タスク優先度設定なし



<After> 適用あり

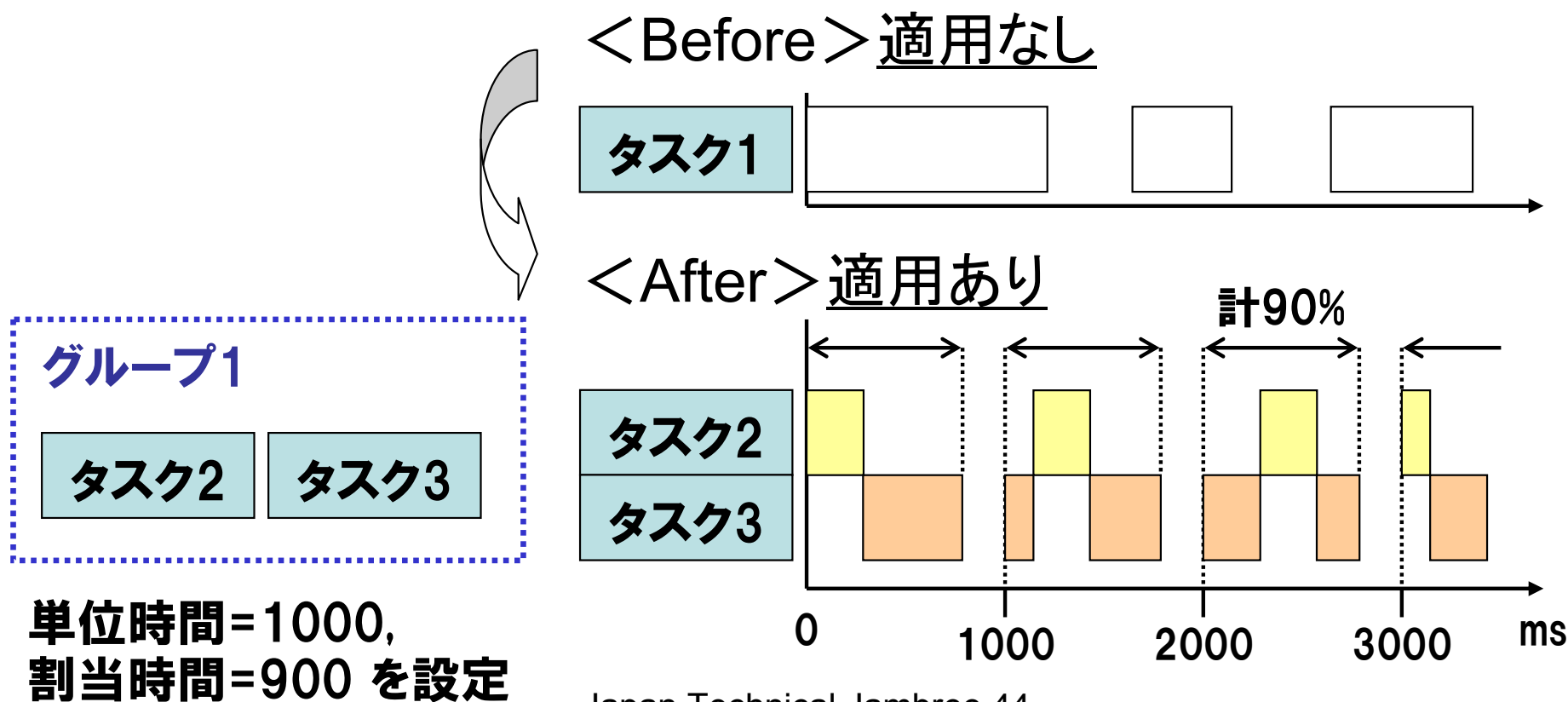


3. AV機器への適用検討

3.2. cpuサブシステム(「RT Group Scheduling」)

⇒Real-Timeタスク(優先度付きタスク)に割り当てる、単位時間当たりのCPU使用時間の制限を可能にする機構。

⇒高優先度タスクの暴走を阻止する適用方法が可。



3. AV機器への適用検討

3.3. AV機能に対する優先的リソース配分の適用案

- (1)cpuサブシステム:AV機能アプリと第三者アプリ(ゲーム等)間でのCPUリソース配分の優先制御に適用
- (2)cpusetサブシステム:AV機能と第三者アプリの同時動作時におけるアプリ実行CPUの振り分け(負荷分散)に適用
- (3)memoryサブシステム:AV機能と第三者アプリ間の総メモリ使用量配分の優先制御に適用
- (4)blkioサブシステム:録画コンテンツの再生処理、及び蓄積処理(ファイルI/O)の優先制御に適用
- (5)net_prioサブシステム:映像ストリームの受信処理(IPTV等)の優先制御に適用

3.4. リソース配分定義の動的切替え

⇒AV機器の機能の多様化を勘案すると、ユーザの利用シーン毎に動作する機能セットが異なるなら、状態(いくつかの利用シーン)に応じて適切なリソース配分定義を複数用意

- ・「見る」「録る」「残す」等の機能の多様化
- ・放送/通信/メディア等のコンテンツ種別の多様化

⇒利用シーンに応じてリソース配分定義を切替えることで(定義ファイルを再読み込みする)、最適なリソース配分が動的に可能

4. Android環境での適用検証

4.1. AndroidカーネルでのCgroups関連コンフィグ設定

⇒Android4.X用カーネルは、Linux 3.0.X以降であり、
コンフィグ設定により、Androidでもそのまま利用可

| Cgroups関連のコンフィグ設定項目 |
|--------------------------|
| CONFIG_CGROUPS=y |
| CONFIG_CGROUP_FREEZER=y |
| CONFIG_CGROUP_DEVICE=y |
| CONFIG_CPUSETS=y |
| CONFIG_PROC_PID_CPUSET=y |
| CONFIG_CGROUP_SCHED=y |
| CONFIG_BLK_CGROUP=y |

4. Android環境での適用検証

4.2. Android上でのCgroupパッケージ移植試行

⇒今回、Cgroupソースパッケージ(libcgroup)を入手し、Android上でビルドを実行。Android.mkへの書き換えは行なったが、標準Cライブラリ(Bionic)へのリンクエラー発生し断念(必要モジュール不足が要因)。

☆成功事例あれば、情報をご提供下さい

⇒そこで、必要最低限の以下の機能モジュールを自作。

①Cgconfigサービス(リソースグループ定義ファイルに基づく、リソースグループの自動生成と管理)

②Cgredデーモン(リソース配分ルール定義ファイルに基づく、タスクの特定グループへの移動実行)

4. Android環境での適用検証

4.3. リソースグループ状態表示ツールの試作

⇒リソースの配分やタスク移動の定義を決め易くするため(系統的に調整・決定できるようにするため)リソースグループ単位のカレント状態を可視化するツールを製作(Javaプログラム)。

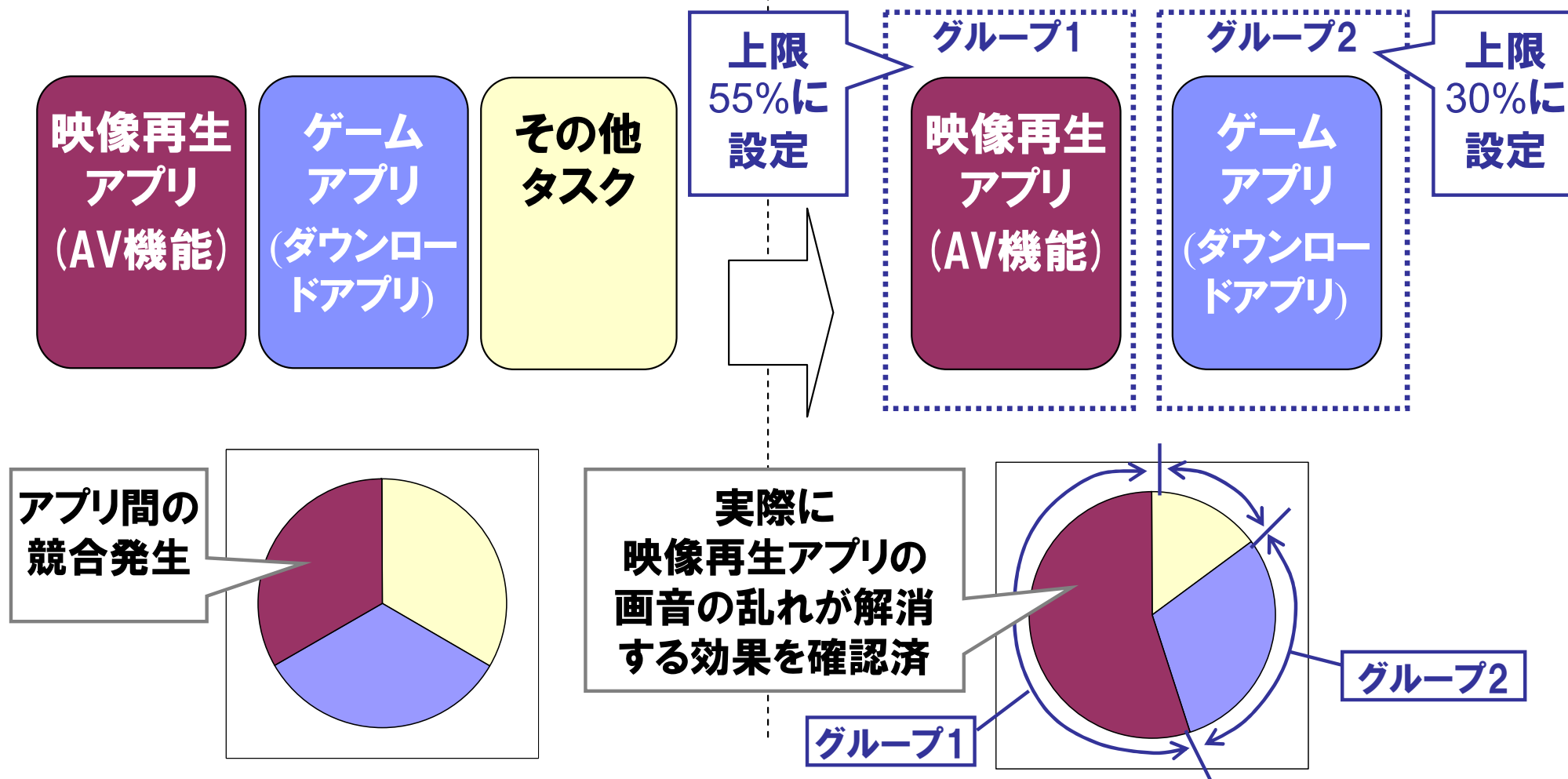


4. Android環境での適用検証

4.4. 映像再生アプリと第三者アプリ間のリソース配分適用

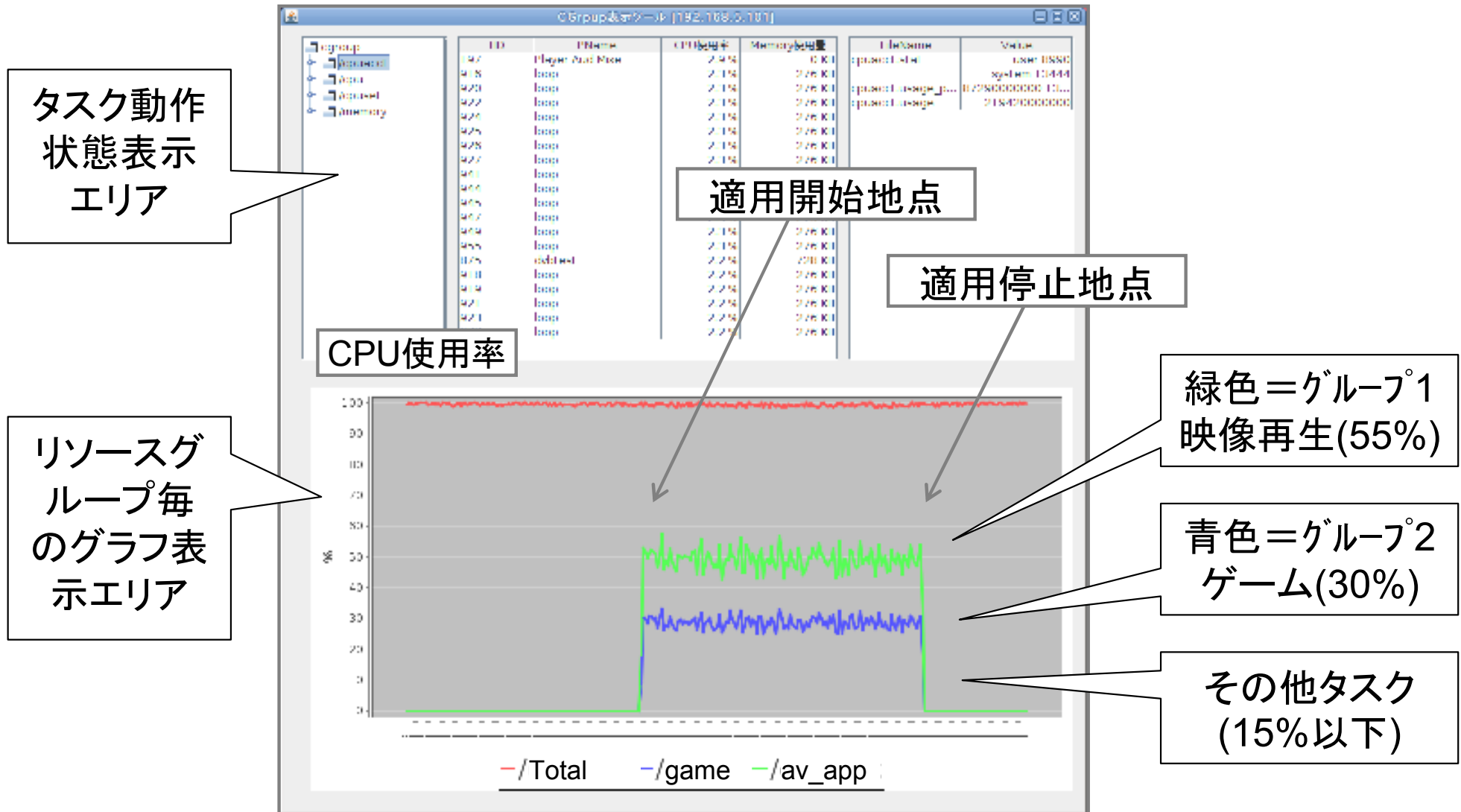
<Before> 適用なし

<After> 適用あり



4. Android環境での適用検証

4.5. リソースグループ状態表示ツールの表示例



- ◆Android環境のAV機器へのリソース管理機構の適用を試行し、映像再生アプリでの効果(安定動作)を検証
- ◆リソース配分定義を系統的に実行可能にするため、リソースグループ状態表示ツールを製作、有用性を確認
- ◆今後の予定
 - (1)AV機器に対する、より複雑な複合リソース配分ケースでの適用試行
 - (2)AV機器における複数のリソース配分定義の動的切り替え方式の検証
 - (3)リソースグループ状態表示ツールの拡張